REMARKS

The Office Action dated July 30, 2003 has been received and carefully noted. The following remarks are submitted as a full and complete response thereto. Claims 1-15 are pending in the present application. Claims 1-15 stand rejected. Reconsideration and allowance of all pending claims are respectfully requested in view of the following remarks.

INFORMATION DISCLOSURE:

Applicant thanks the Examiner for acknowledging consideration of the references accompanying the Information Disclosure Statement filed on August 30, 2001.

CLAIM REJECTIONS:

35 U.S.C. § 102

Claims 1-15 are rejected under 35 U.S.C. § 102 as being anticipated by U.S. Patent 5,745,889 to Burrows ("*Burrows*"). Applicant respectfully traverses this rejection for the following reasons.

Applicant's invention is directed to a method and apparatus for a table lookup index that provides improved access to a table such as an addressing table for transmission of data packets in a network switch. (Specification, pg.1, par. 7). Conventionally, table entries for these switches are indexed using a linear index system. Referring to Fig. 1C of the instant application, a linear index of this type may reference a 64K table with indices such as I(1), I(2), I(3) and so on. Each index is linearly indexed

into the 64K table giving it a one to one correspondence between the index and table entry. (Specification pg. 2, par. 13). Accordingly, if indices each I(1), I(2), I(3)....are 16 bits long an entry in the table can be found in 16 clock cycles using standard binary searching. However, when a new address must be learned by inserting an entry or deleting an entry, it will take a long time to insert the new address to the table because the index must be sorted after each insertion/deletion. (Specification pg.2 par. 13). The sorting time correlates to the switching time and increases with the size of the table.

The present invention addresses this issue by using a data structure (e.g., 200; Fig. 2A) which includes an index segment I, a bucket segment N and a data segment M. (Par. 41). Index segment I is linearly indexed to a bucket segment N, the combination of which directly selects an entry in the table. Because the table is segmented into sublevels of buckets N(#) which are indexed by the index segment I(#), an insert or delete function can be dependent on the bucket size (which can be programmed to fit a particular need), as opposed to the table size. This arrangement thus improves the efficiency of table updating and hence switching speeds.

Burrows

While *Burrows* addresses parsing and indexing a database, in contrast to Applicant's invention, *Burrows* discloses a method for parsing an extremely large database (e.g., the Internet) having content (e.g., web pages) and content attributes (e.g. location). (Col. 1, ll. 50-52). *Burrows* discloses a parsing module 30 (Fig. 2) which breaks down web pages 200 into fundamental indexable elements including the words

found on a web page and the location of these words on the web page. These parsed indexable elements are referred to as "atomic pairs" 400 which consist of a word (410) and its location (420). (Col. 4, ll. 51-57). The word is a literal representation of the parsed portion of the web page content and the location is a numeric value (e.g., assigned integer). (Col . 4, ll. 57-64).

Applicant respectfully submits that *Burrows* fails to teach or suggest the claimed *method of performing a table look-up*. *Burrows* fails to teach or suggest any type of search or database query including the steps of: *receiving data through an input source, parsing the received data into an index portion and a corresponding bucket portion, indexing the index portion to the corresponding bucket portion* and/or *accessing table information stored in a look-up table*. (claim 1).

To the extent Applicant comprehends its reasoning, the Office Action cites various passages in *Burrows* in an attempt to reconstruct Applicant's claims in a piecemeal fashion based on improper hindsight of Applicant's specification.

First, it is respectfully submitted that *Burrows* generally relates to creation of and updating of database records, as opposed to the method of table look-up claimed by Applicant. As shown in Fig. 2 of *Burrows*, an actual query 52 (e.g., look-up) from an Internet user entirely avoids the parsing 30 and indexing 40 loop cited by the Office Action as anticipating the steps of Applicant's claimed.

Furthermore, the Office Action alleges that *Burrows* discloses *parsing the data into an index portion* (citing col. 4, ll. 51-56) *and a corresponding bucket portion*, (citing

col. 7, ll. 33-39). Presumably, the Office Action reasoning alleges that word location pairs 400 (which results from the parsing operation 30) comprises an *index portion* as location 420 and a *bucket portion* as word 410. If however, this is not what is alleged, the Examiner is requested to issue a new Office Action to clarify what elements of the *received data* that *Burrows* discloses are parsed into the index portion and the bucket portion. The intent of the Office Action is further made unclear where in the footnote on page 3 the Office Action alleges that both word 410 and location 420 are a bucket. ("Examiner interprets the steps 410 and 420 as a bucket.")

Applicant respectfully requests the Examiner to clarify that if 400 in *Burrows* represents the parsed bucket (i.e., 410 and 420), which element in *Burrows* represents the parsed index portion?

Notwithstanding, *Burrows* discloses parsing the web pages into the word location pairs 400 which includes both a word 410 and a location 420.

Next the Office Action alleges that the claimed step of "*indexing the index portion to the corresponding bucket portion*" is disclosed at column 5, lines 2-15. To the contrary, this passage of *Burrows* discloses sorting the pairs 400 first in word order (e.g., alphabetically) and then in location order. The sorted pairs 400 are then used to generate an index 70 of the words of the web pages 200. Respectfully, the *Burrows* sorting and generating an index 70, is not at all analogous to the claimed "*indexing said index portion to said corresponding bucket portion.*

"Indexing," in the context of Applicant's claim 1 relates to locating or matching using an index, as opposed to generating an index 70 which is described in the cited portions of *Burrows*. Furthermore, as interpreted earlier in the Office Action, where the claimed "*index portion*" comprises the location (420) of pair 400 and the "*bucket portion*" comprises the word (410) of parsed pair 400, the allegation set forth in the Office Action would mean that the location (420) (i.e., the alleged *index portion*) is indexed to the word (410) (i.e., the alleged *bucket portion*). This is clearly is not the case as *Burrows* discloses that word (410) and location (420) coexist as a pair (400) throughout the process and are in fact compressed in index 70 as part of data structure 71. (Col. 5, ll. 4-6).

Additionally, *Burrows* does not disclose *accessing table information stored in a look-up table using the bucket portion*. The Office Action alleges that this is disclosed by *Burrows* at column 5, lines 49-56 where *Burrows* discloses a user querying 52 the index 70. However, it is respectfully noted that this portion of *Burrows* is not related to the parsing and indexing and thus is not part of a "method of performing a table look-up which includes receiving, parsing and indexing as claimed by Applicant.

The discrepancies in the Office Action reasoning result from the fact, that in the present invention, a table look-up is performed as opposed to building a database index for an Internet search engine, which is what is disclosed by *Burrows*.

In present invention the received and parsed index portion is indexed (or matched) to the corresponding bucket portion and the bucket portion points to a data entry stored in

the lookup table (if there is a table entry for the corresponding bucket). In this manner, the look up table can be searched as well as updated, without having to search/update the entire table, but only the table portions corresponding to the bucket portions. Because *Burrows* does not disclose the method of performing a table look-up as recited in Applicant's claim 1, claim 1 and dependent claims 2-7 are not anticipated or rendered obvious by *Burrows*.

In respect to claim 8, for the same reasons discussed above, Burrows does not teach or suggest *a data parser that pareses received data into an index portion and a corresponding bucket portion.* Further, *Burrows* does not teach or suggest *an indexer that indexes the index portion to the bucket portion.* Further, *Burrows* fails to teach or suggest *a lookup device that accesses a look-up table using the corresponding bucket portion.*

In respect to claim 15 it is evident that *Burrows* does not teach or suggest *a network switch having multiple ports used for receiving or exporting data, each of the multiple ports connected to one another. Burrows* further fails to teach or suggest *multiple address resolution logic (ARL) devices each being connected to one of the multiple ports.* Accordingly, *Burrows* could not possibly disclose the claimed ARL device including the parser, indexer and/or look-up device as recited in claim 15. In fact, because the Office Action does not address the features recited in claim 15 in any manner whatsoever, the rejection of claim 15 is improper on this basis alone.

In respect to the remaining Office Action points regarding Applicant's dependent claims, Applicant does not separately address each Office Action comment but traverses the rejection of the dependent claims on the basis that they are patentable over *Burrows* for the same reasons previously discussed with respect to independent claims 1, 8 and 15. The Office Action misinterpretation of *Burrows* merely increases in the discussion of the dependent claims. For example, the Office Action cites Fig. 3, col. 7, ll. 13-61, of *Burrows* as disclosing the claim 2 recitation of *wherein the step of indexing the index portion to the bucket portion is the step of linearly indexing the index portion to the bucket portion.*

Respectfully, *Burrows* in no way suggests that locations 420 (alleged index portion) are <u>linearly indexed to</u> words 410. (alleged bucket portion). As discussed previously, the word 410 and location 420 are stored together in pairs 400. (Col. 7, ll. 15-16) .

Further, the Office Action cites col. 7, ll. 33-40 of *Burrows* as disclosing that the bucket portions are binary sorted. In fact, this recited passage of *Burrows* merely mentions that the words 410 may be expressed <u>as a binary endcoding of an ASCII value</u> and that the locations 420 incrementally increase by one for each word parsed. This is not the same however as binary sorting which <u>sorts</u> based on a binary value.

For all the reason discussed above, Applicant submits that *Burrows* does not anticipate or render obvious, claims 1-15. Accordingly, the Examiner is respectfully requested to reconsider and withdraw the §102(b) rejection based on *Burrows*.

If for any reason the Examiner determines that the application is not now in condition for allowance, it is respectfully requested that the Examiner contact, by telephone, the undersigned attorney at the indicated telephone number to arrange for an interview to expedite the disposition of this application.

In the event this paper is not being timely filed, Applicant hereby petitions for an appropriate extension of time. Any fees for such an extension together with any additional fees or deficiency of fees may be charged to Counsel's **Deposit Account 50-2222**.

Respectfully submitted,

Kevin F. Turner
Registration No. 43,437

Customer No. 32294
SQUIRE, SANDERS & DEMPSEY LLP
14^{TH} Floor
8000 Towers Crescent Drive
Tysons Corner, Virginia 22182-2700
Telephone: 703-720-7800
Fax: 703-720-7802